IDACT Query Manager for Heterogeneous Dataset Assimilation

Brian Hay

Kara Nance

University of Alaska Fairbanks

IDACT Overview

- Goal of IDACT is to allow data consumers to access data from multiple sources in a format that meets their needs, without the need for technical knowledge of the data location, format, or access method.

IDACT Overview

- Each IDACT instance is deployed by an organization for a particular subject domain.
  - For example, a university could run an IDACT instance for the internal and external geophysical data sources it uses.
  - Subject domain can be as general or specific as necessary.
  - Instance usage by data consumers can be restricted or open.

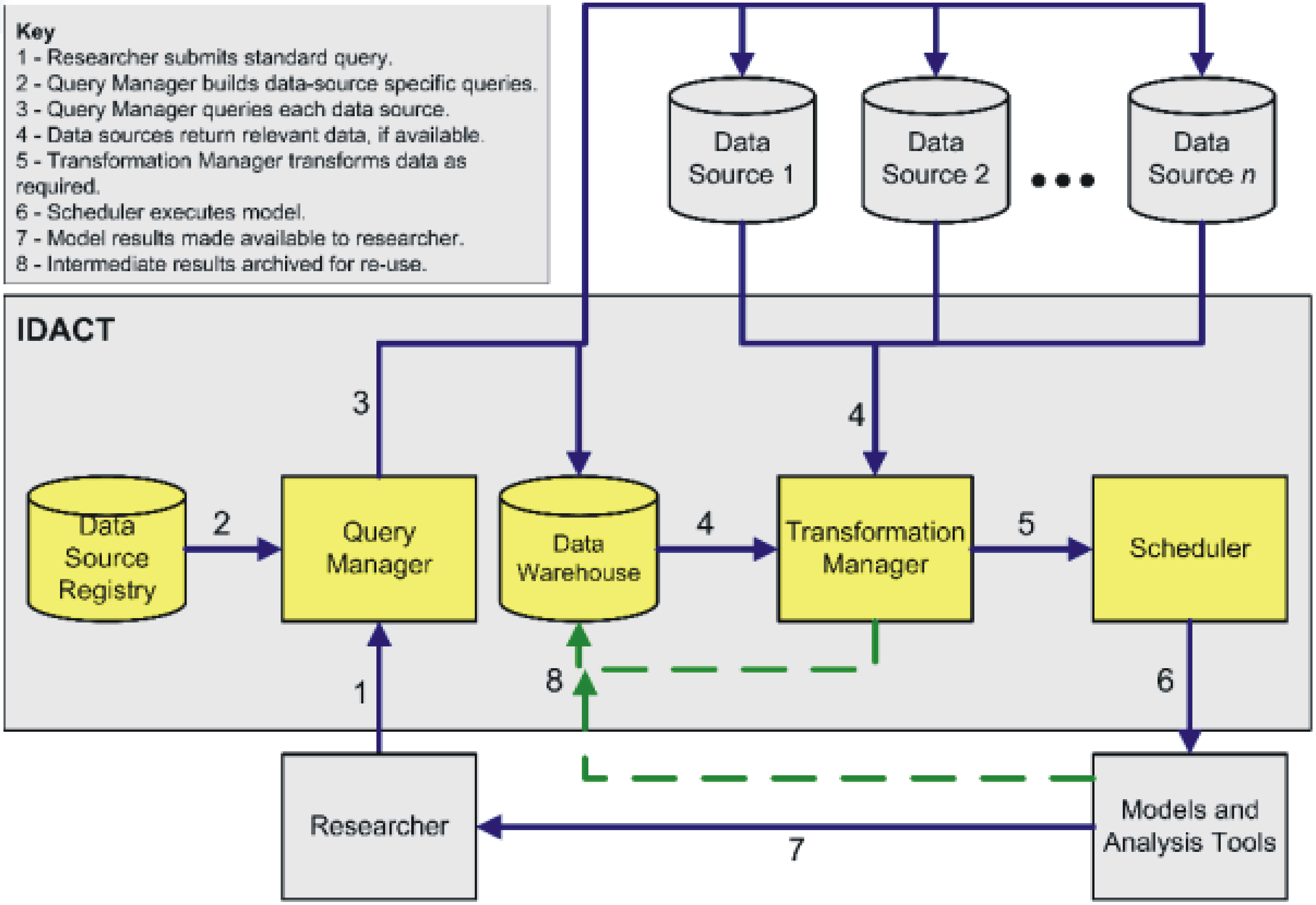
IDACT Overview

- Data Consumers
  - Typically scientists or researchers in this context.
  - Want to be able to access data necessary for their research, without spending money or time on data acquisition and conversion issues.

IDACT Overview

- Data Owner
  - Usually either a researcher who produced a dataset, or the administrator of the dataset storage system.
  - Data Owner submits a data source to an IDACT instance, and the data is then available to data consumers.

IDACT Overview



Key

1 - Researcher submits standard query.

2 - Query Manager builds data-source specific queries.

3 - Query Manager queries each data source.

4 - Data sources return relevant data, if available.

5 - Transformation Manager transforms data as required.

6 - Scheduler executes model.

7 - Model results made available to researcher.

8 - Intermediate results archived for re-use.

Problem Statement

- Query Manager (QM) builds ‘queries’ to acquire data from data sources.
- In order to perform this task, the QM must be able to determine which data sources store the data relevant to the data consumer’s request.

Problem Statement

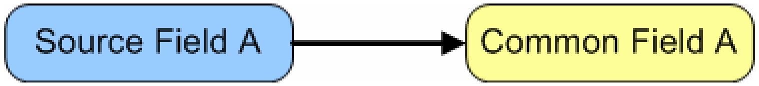
- Transformation Manager (TM) builds new transformations if necessary to produce data in a format that meets the needs of the data consumer.
- In order to perform this task, the TM must be able to determine which components of the data to transform, and in what manner they should be transformed.

Problem Statement

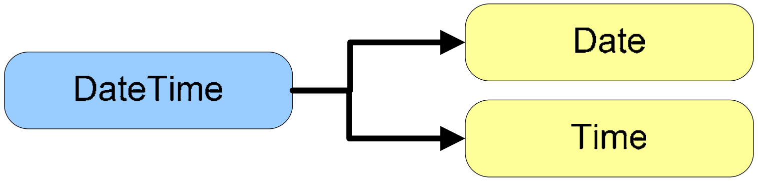
- The Datasource Registry (DR) provides this functionality for the QM and TM.
- The DR stores a description of a datasource which includes associations between datasource fields and “common fields”.

Problem Statement

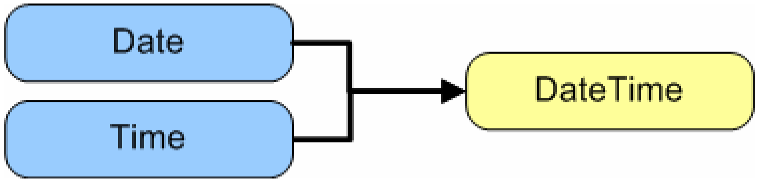
- The three association types are:
  - Simple: one source field maps to one common field.



  - Split: one source field maps to multiple common fields.



  - Combine: multiple source fields map to one common field.



Problem Statement

- DR API allows the QM and TM to request association information.
- The problem lies in how to allow a data owner to easily add a new datasource to the DR (i.e. how to populate the DR with new associations).

Association Search

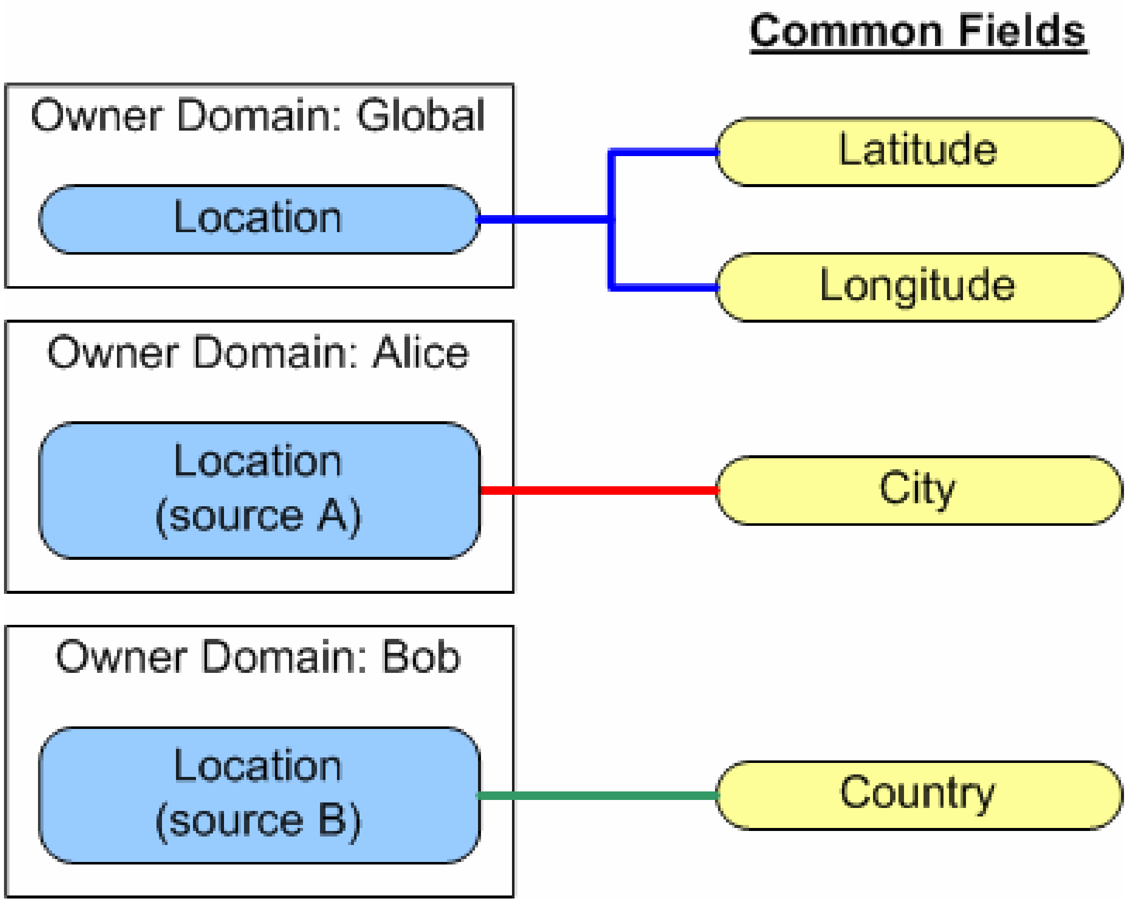
- First approach relies on a search of existing associations.
- For each field name in the datasource, find any associations currently defined in the DR.
- Limited to datasources which have named fields (quite common).

Association Search

- Associations are organized by datasource and owner domains.
- Search gives preference to associations in same owner domain as submitter.
- Search order is owner domain of submitter, then global owner domain, then any additional owner domains.
- Result of search is an ordered list of likely associations.

Association Search

- Suppose Alice submits a new datasource C which includes a field named “Location”.
- *Alice* owner domain is searched first, and association to common field “City” is added to the list.
- *Global* owner domain is searched next, and split association to common fields “Latitude” and “Longitude” is added to the list.
- *Bob* owner domain is searched last, and association to common field “Country” is added to the list.



Association Search

- The result of the example search is:
  - {City}, {Latitude, Longitude}, {Country}
  - The first item in the list is chosen as the most likely association, which Alice can accept or reject.
  - If she rejects the proposed association, then the rest of the list is presented as likely associations.
  - Alice has complete control over the association process, and can even create new common fields if necessary.

Association Search

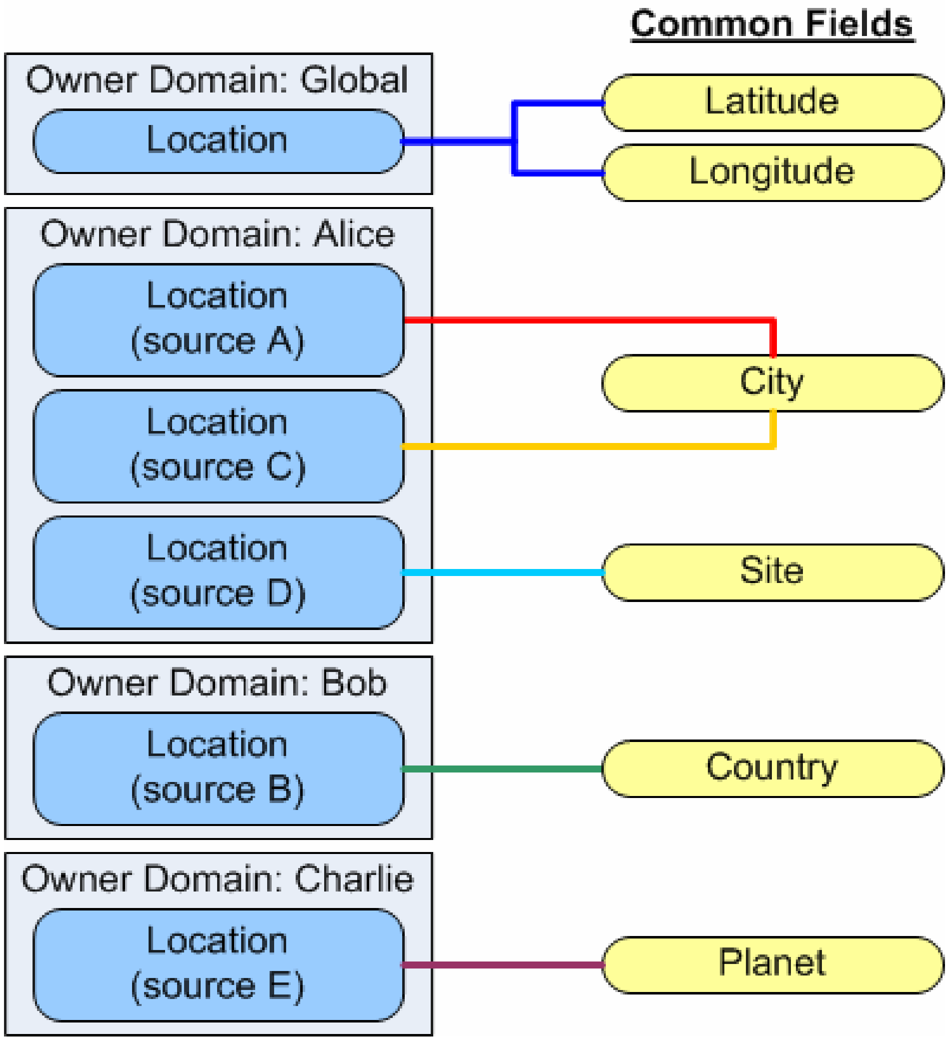
- Once a data owner decides on an association, it is added to the DR, and can be used by the QM and TM.
- The new association is also used for future data submissions, so the field mapping process improves with each new datasource.

Conflict Resolution

- It is not uncommon to find multiple associations in each of the three stages of the search.
- As a result, there must be a conflict resolution strategy so that an ordered list can be produced.

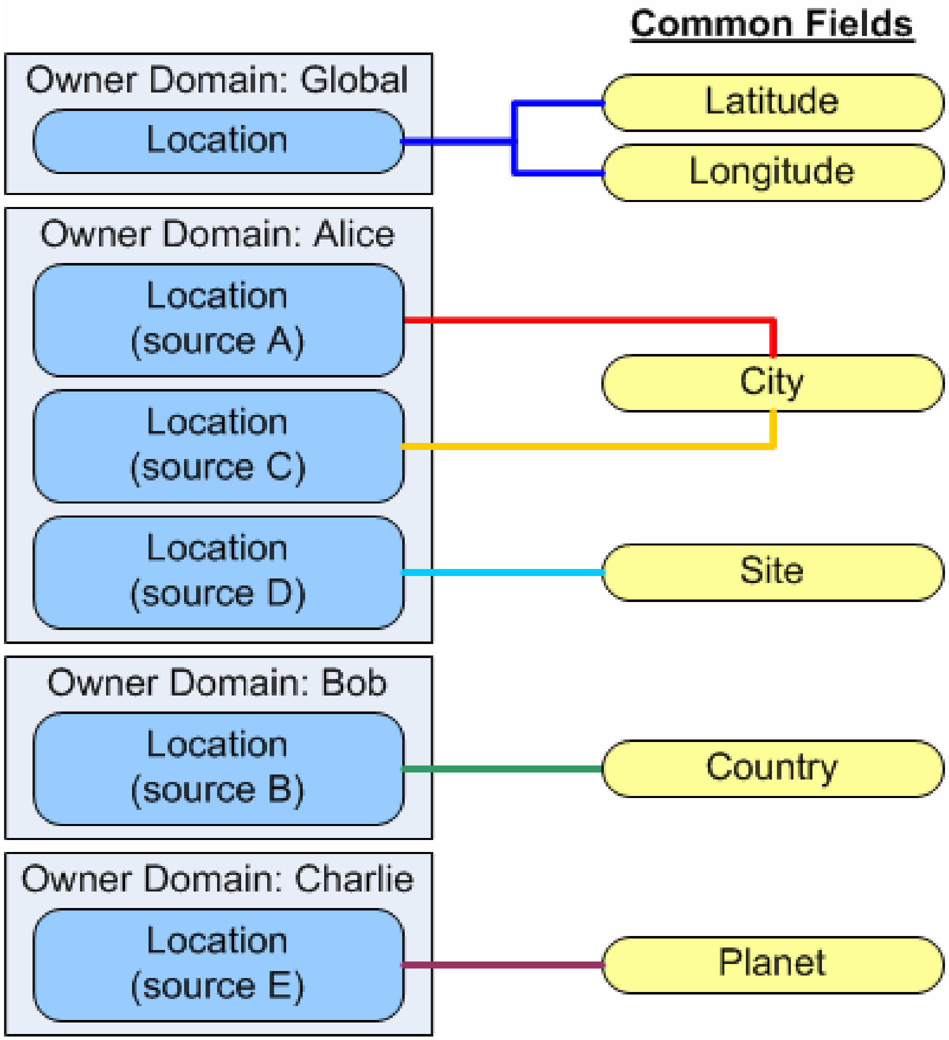
Conflict Resolution

- Suppose that the associations for the "Location" source field are now as shown.
  - Context is used first to resolve conflicts, if possible.
  - Preference is then given to the association which appears most frequently.
  - If neither context nor frequency resolves the conflict, preference is given to the most recently created association.



Conflict Resolution

- Alice now submits a new datasource
  - In the *Alice* Owner Domain (OD), there is a conflict, which is resolved using frequency to give preference to “City” over “Site”.
  - In the *Global* OD there is no conflict.
  - There is a conflict between the associations to “Country” in the *Bob* OD, and to “Planet” in the *Charlie* OD. This is resolved in favor of “Planet”, since this was the most recently created.

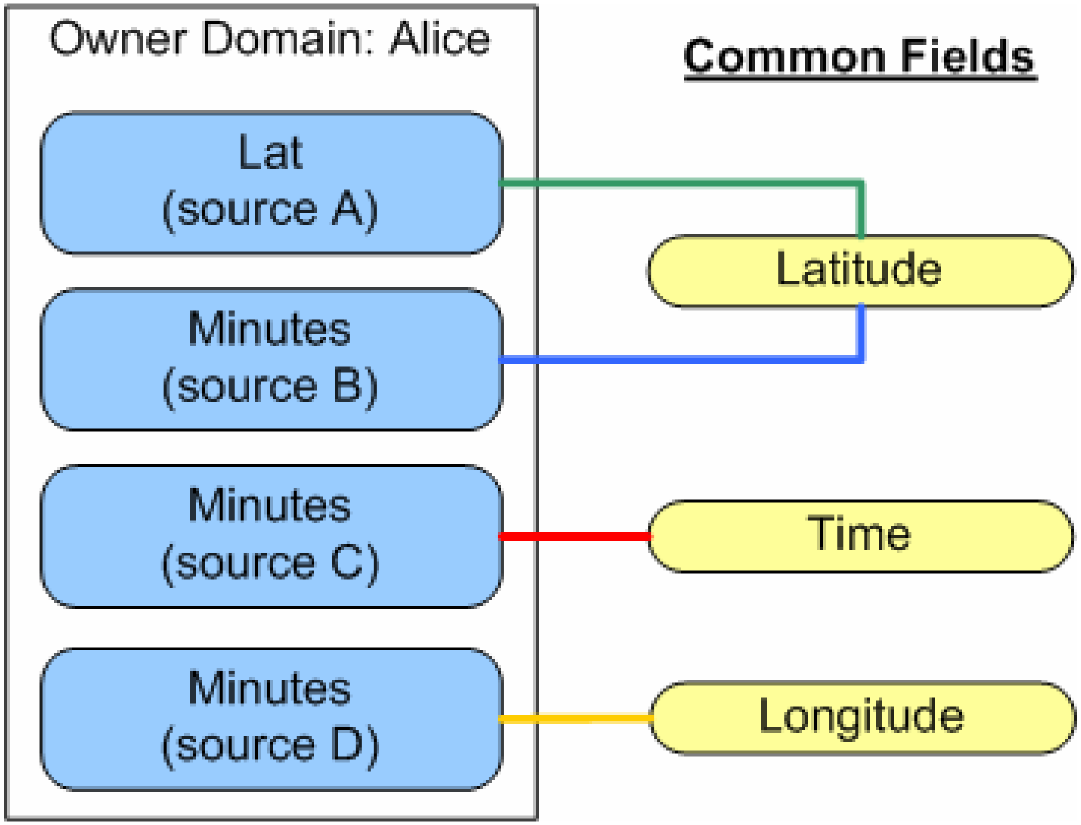


Context

- Context can be applied to datasources which have hierarchical (or partially hierarchical) organization.
  - Context can be expressed in terms of commons fields.
  - Can be useful for conflict resolution.

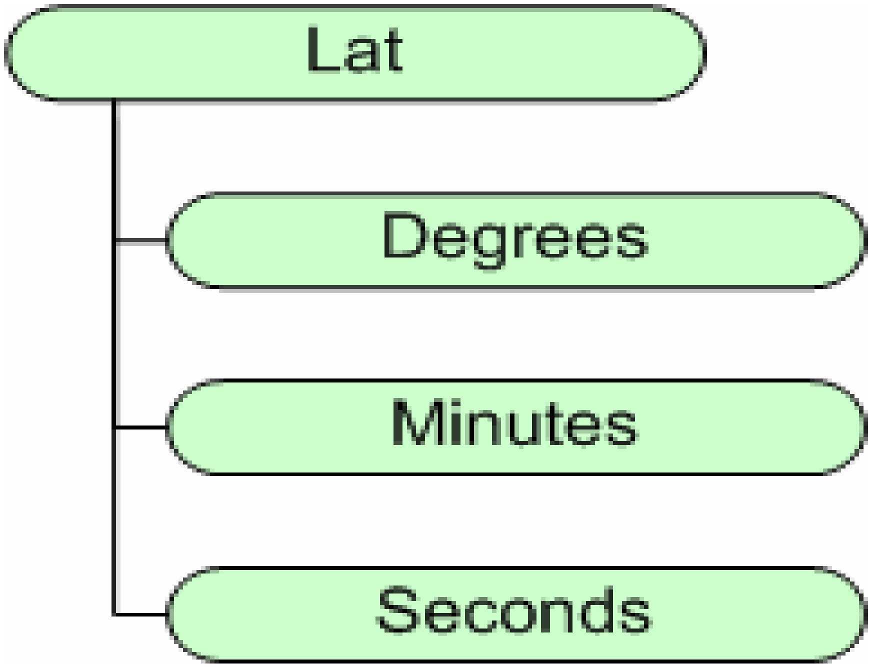
Context

- Suppose a field named *minutes* is encountered in datasource *E*.
  - Three candidate associations are found in the *Alice* owner domain.
  - Which of these associations should be given preference?
  - Context may help determine the most likely association.



Context

- By viewing the minutes field from datasource *E* in context, preference can be give to an association with the *Latitude* common field.



Partial String Matching

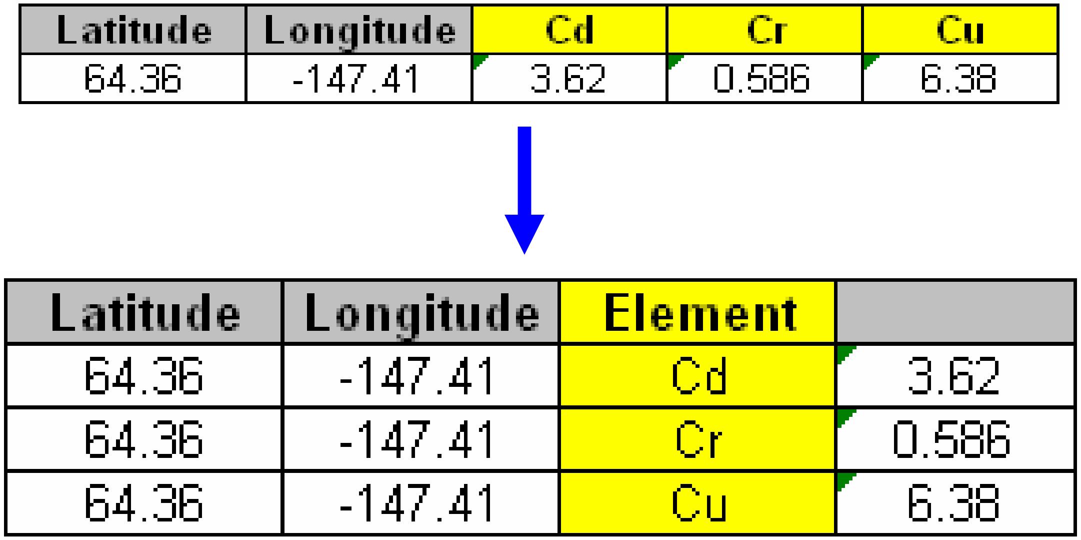
- Partial field name matching can be effective in identifying potential associations.
  - For example, a source field named *Measurement_Date* may not result in any matches.
  - However, a potential association could be found as a result of a partial match with the *Date* common field.
  - Used successfully in the SIMON agent.

Field Values

- Field values are also useful in finding potential associations.
    - Patterns can be used to find potential associations.
    - For example, regular expressions are used to find likely matches.

Field Names and Values

- Field **names** may also be compared against patterns or look-up tables of common field **values** to reorganize the data.



Latitude	Longitude	Cd	Cr	Cu
64.36	-147.41	3.62	0.586	6.38

Latitude	Longitude	Element	
64.36	-147.41	Cd	3.62
64.36	-147.41	Cr	0.586
64.36	-147.41	Cu	6.38

Conclusion

- Basic objectives are
  - Attempt to find reasonable candidate associations automatically.
  - Use candidate association lists to assist the data owner during the data submission process.

Conclusion

- The process improves with use, as the DR learns from past submissions and can provide more meaningful candidate associations.